1. Choose the correct answer for the following questions: [7 points]

a- Consider the following five functions:

$$f_1 = 10^n, f_2 = n^{1/3}, f_3 = n^n, f_4 = \log(\log n), f_5 = 2^{\sqrt{\log n}}$$

Then, arrange the above functions in ascending order.

i. f_4, f_5, f_2, f_1, f_3

ii. f_5, f_4, f_2, f_1, f_3

iii. f_4, f_5, f_1, f_2, f_3

iv. f_5, f_4, f_2, f_3, f_1

b- What is the average case time complexity of recursive insertion sort?

i. O(n)

ii. O(n log n) -1

iii. O(n2)

iv. O(log n)

c- Which of the following sorting algorithm is in place?

i. Recursive insertion sort

ii. Merge sort

iii. Radix sort

iv. Counting sort

d- What is the ~~typical~~ running time of a quick sort algorithm?

i. O(N)

ii. O(N log N)

iii. O(log N)

vi. $O(N^2)$

e. Master's theorem is used for?

i. Solving recurrences ✓

ii. Solving iterative relations

iii. Analysing loops

vi. Calculating the time complexity of any code

f. Under what case of Master's theorem will the recurrence relation of merge sort fall?

$T(n) = 2T(\frac{n}{2}) + n$

$a = 2,\ b = 2,\ f(n) = n$

$n^{\log_b a} = n = n$

i. 1

ii. 2 ✗ – 1

iii. 3

vi. It cannot be solved using master's theorem

g. We can solve any recurrence by using Master's theorem.

i. true

ii. false ✓

2. Using the definition of θ, find $g(n)$, C_1, C_2, and n_0 in the following: [6 points]

$$7n^6 - 3n^4 + 2n^3 + n^2 - 2 \in \theta(g(n))$$

big O:

$7n^6 - 3n^4 + 2n^3 + n^2 - 2 \le 15n^6$

$C_1 = 15$

$n_0 = 1$

big O:

$7n^6 - 3n^4 + 2n^3 + n^2 - 2 \le 15n^6$ ✓

$C_1 = 15$

$n_{01} = 1$

big Ω

$7n^6 - 3n^4 + 2n^3 + n^2 - 2 \ge 6n^6$

$C_2 = 6$

$n_0 = \frac{8}{7-6} = \frac{8}{1} = 8$

$n_{02} = 1$

big Ω:

$7n^6 - 3n^4 + 2n^3 + n^2 - 2 \ge 6n^6$

$C_2 = 6$

$n_{02} = 8$

$n_{02} = \frac{3+2+1+2}{7-6} = \frac{8}{1} = 8$ ✓

$\Theta(n^6)$, $C_1 = 15$, $C_2 = 6$, $n_0 = \max(n_{01}, n_{02}) = 8$

Problem 2 [9 points] 1/9

a- Consider the following pseudo-code:

```
void SampleFct(int arr[], int n)
{
        if (n <= 1)   1
                return;   T
        SampleFct( arr, n-1 );   n-1
        int key = arr[n-1];   1
        int j = n-2;   1
        while (j >= 0 && arr[j] > key)   n-1   for(j=n-2, j≥0 && arr[j]>key, j--){
        {
     n-2        arr[j+1] = arr[j];
                j--;   n-2
        }
        arr[j+1] = key;   1
}
```

$T(n) = T(n-1) + 3n - 2$

b- Which problem does this algorithm solve (What does it do)?

it will sort the array
by comparing each arr(i) with arr [i+1]
~~by comparing each arr[i] with arr[i+1]~~

after its recursively make the array
smaller at every recurrince until ~~its~~ there is two nodes

Sorting !! -2

c- What is the recursive equation corresponding this pseudo-code.

$$T(n) = T(n-1) + 3n - 2$$

d- Prove the big-Oh performance of the **SampleFct**.

$$T(n) = T(n-1) + 3n - 2 \quad k=1$$

$$T(n) = [T(n-2) + 3(n-1) - 2] + 3n - 2 \quad k=2$$

$$T(n) = [T(n-3) + 3(n-2) - 2] + 3(n-1) + 3n - 2 - 2 \quad k=3$$

$\vdots$

$$T(n) = T(n-k) + 3(n-(k-1)) - 2^{k}$$

we will stop when

$n - k = 1$

$k = n - 1$

$$T(n) = T(1) + 3(n + (n-1) + 1) - 2^{n-1}$$

$$= 3(\cancel{2n}1) + 2 - 2^{n-1}$$

$$= 3$$

?? -2

Problem 3 [9 points] 4/9

Consider the following recurrence relations. Solve each recurrence relation using recursive substitutions and find its asymptotic performance.

(a)

$$T(n) = 4T(\tfrac{n}{5}) + 3n^2$$

$T(n) = 4T(\frac{n}{5}) + 3n^2 \quad K=1$

$T(n) = 4\left[4\,T(\frac{n}{25}) + 3(\frac{n}{5})^2\right] + 3n^2 \quad K=2$

$16\,T(\frac{n}{25}) + 12\,\frac{n^2}{5} + 3n^2$

$T(n) = 16\left[4(\frac{n}{125}) + 3\,\frac{n^2}{25}\right] + 12\frac{n^2}{5} + 3n^2 \quad K=3$

$= 64(\frac{n}{125}) + 48\,\frac{n^1}{25} + 12\frac{n^2}{5} + 3n^2$

$= 64(\frac{n}{5^K}) + 48\,\frac{n^2}{5^{K-1}} + 12\,\frac{n^2}{5^{K-2}}$

$= 64T(1) + n^2\sum_{i=0}^{K-1}\frac{1}{5^i} \leq 64 + \frac{n^2}{4}$

$= 64T(1) + n^2\sum_{i=0}^{K-1}\frac{1}{5^i} \leq 64 + \frac{n^2}{4}$

25 + 25 = 50

25 + 50 = 75

25 + 75 = 100

25 + 100 = 125

we will stop when $\frac{n}{5^i} = 1$

$K = \log_5 n$

O(?) -2

$$T(n) = T\left(\frac{n}{3}\right) + 2n + 5$$

$T(n) = T(\frac{n}{3}) + 2n + 5$ $k=1$

$T(n) = [T(\frac{n}{9}) + 2\frac{n}{3} + 5] + 2n + 5$ $k=2$

$T(n) = [T(\frac{n}{27}) + 2\frac{n}{9} + 5] \frac{2n}{3} + 2n + 5 + 5$ $k=3$

$= T(\frac{n}{27}) + \frac{2n}{9} + \frac{2n}{3} + 2n + 5 + 5 + 5$

$T(n)$ $T\frac{}{}$

$T(\frac{n}{3^k}) + \frac{2n}{3^{k-1}} + \sum_{i=1}^{k-1} 5$

-3

we will stop when

$\frac{n}{3^k} = 1$

$k = \log_3 n$

$T(1) + \frac{2n}{3^{\log n - 1}} + (\log n - 1)(5)$

$+ \frac{2n}{3^{-1} n} + 5\log n - 5$

$\frac{2}{3^{-1}} + 5\log n - 5$

Problem 4 [4 points] 3/4

Solve the following recurrence using the Master theorem by giving tight θ-notation bounds. Justify your answers.

(a) $T(n) = 4T(n/2) + 3n\, log(n)$

$a = 4 \quad , b = 2 \quad , c = 3n\log(n)$

$n^{\log_b a} = n^{\log_2 4} = n^2 > 3n\log(n)$

case 2

$\Theta(n^2)$

??

-1

Problem 5 [15 points] 00/15

(a) Describe an in-place algorithm that takes as input an array of n integers and rearranges it so that all even integers come first and then all odd integers come next. You are allowed to use ONLY a constant amount of extra storage.

[10 points]

S(Arr[],n){

(b) What is the time complexity of your algorithm? You need to show the step count and Big_Oh estimate.

[5 points]